PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-091449
(43)Date of publication of application : 10.04.1998

(51)Int.Cl. G06F 9/44
G06F 9/06

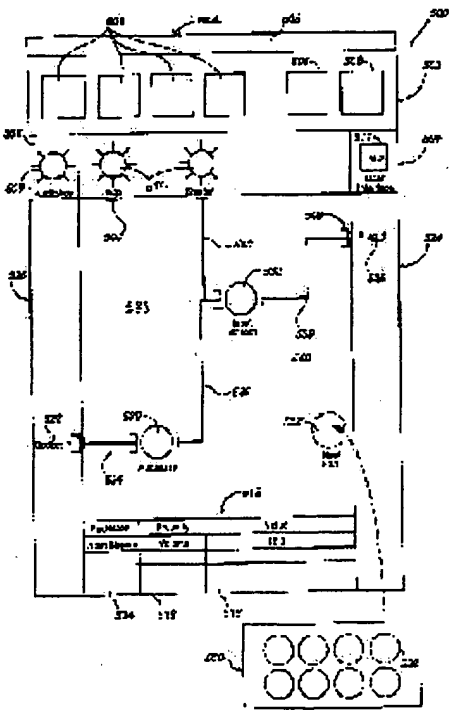
(21)Application number : 09-178698
(22)Date of filing : 03.07.1997

(71)Applicant : SUN MICROSYST INC
(72)Inventor : FOWLOW BRAD G
NUYENS GREG B
LUDOLPH FRANK

(30)Priority
Priority number : 96 675850 Priority date : 03.07.1996 Priority country : US

(54) VISUAL ASSEMBLING TOOL FOR CONSTITUTING APPLICATION PROGRAM BY USING DISTRIBUTED OBJECT ON DISTRIBUTED OBJECT NETWORK

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce a software developing time by providing catalogue facility including reference to a pre-existed object, linking reference so as to provide application structure environment and permitting reference to be used to application structure environment.
SOLUTION: A component like the component 552 of a component catalogue 550 is selected and the component is drugged from the component catalogue to a work sheet. Then, the component is deformed to component parts 552'. The plug and the socket of the parts in the work sheet are connected to each other, so that a composition builder generates a code corresponding to the establishment of required connection between the parts. Thus, the composition builder refers to the code to be used in a distributed system and provides a method being more simple than that for reusage.



LEGAL STATUS
[Date of request for examination]
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平10－91449

(43)公開日　平成10年(1998)４月10日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	9/44	５３０	Ｇ０６Ｆ	9/44	５３０Ｐ
	9/06	５３０		9/06	５３０Ｗ

審査請求　未請求　請求項の数24　ＯＬ　（全 19 頁）

(21)出願番号　特願平9－178698

(22)出願日　平成９年(1997)７月３日

(31)優先権主張番号　０８／６７５８５０

(32)優先日　1996年７月３日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　595034134
サン・マイクロシステムズ・インコーポレイテッド
Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，　Ｉｎｃ．
アメリカ合衆国　カリフォルニア州　94303　パロ　アルト　サン　アントニオ　ロード　901

(72)発明者　ブラッド　ジー．　フォウロウ
アメリカ合衆国，　カリフォルニア州，　レッドウッド　シティ，　ヒルサイド　ロード　546

(74)代理人　弁理士　長谷川　芳樹　（外５名）

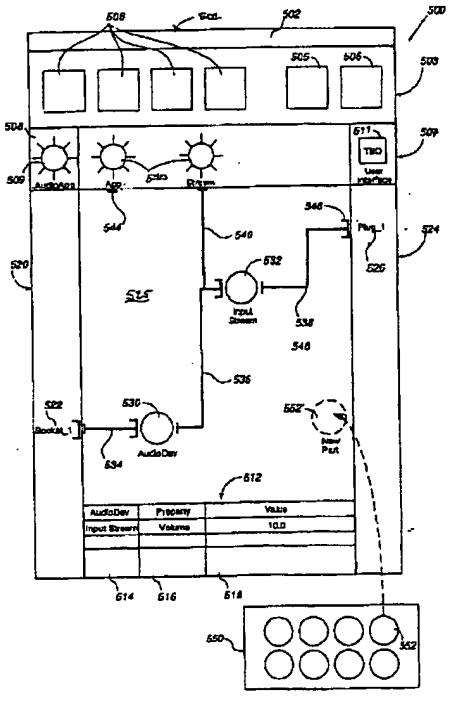
最終頁に続く

(54)【発明の名称】　分散オブジェクト・ネットワーク上で分散オブジェクトを利用してアプリケーションプログラムを構成するための視覚的組立ツール

(57)【要約】

【課題】　すでに開発した分散オブジェクトを再利用して新たなアプリケーションを開発するために、コンポーネントのカタログを見てコンポーネントを選択しリンクを作成することを支援するアプリケーション構築環境が必要となる。

【解決手段】分散オブジェクトシステム上に供給された既存オブジェクトへのリファレンスを提供するコンポーネントを含むカタログファシリティを提供するステップから構成される。カタログファシリティからアプリケーションプログラムに含めるためにコンポーネントが選択される。部品は、選択されたコンポーネントから利用し、オブジェクト指向アプリケーションソフトウェアのための、コンピュータコードによって提供される関係を定義する部品間のリンクを選択・定義するための機能を利用するこのシステム上の既存オブジェクトを参照する少なくとも一つ以上の部品へリンクを作成できる上記環境で利用できる。



【特許請求の範囲】

【請求項１】　分散オブジェクトシステム上にインストールされるオブジェクト指向アプリケーションソフトウエアを構築するためのコンピュータにインプリメントされた（computer-implemented）方法であって、オブジェクト指向アプリケーションソフトウェアはアプリケーションプログラムインタフェースと少なくとも一つの既存のオブジェクトまたは既存のオブジェクトの派生物（derivative）を含み、

ａ）分散オブジェクトシステム上の既存オブジェクトのインプリメンテーション（implementation）へのリファレンスを含むカタログ・ファシリティ（catalog-facility）を提供するステップと、

ｂ）前記リファレンスをリンクし（link）それらの間の関連を定義するアプリケーション構築環境を提供し、前記オブジェクト指向アプリケーションソフトウェアのためのコンピュータコードを提供するステップであって、前記アプリケーション構築環境は前記リファレンス間のリンクを選択および定義するためのファシリティ（facility）を含み、

ｃ）前記カタログファシリティから前記アプリケーションソフトウェアに含めるためのリファレンスを選択する選択ステップと、

ｄ）前記既存のオブジェクトへの前記リファレンスを前記アプリケーション構築環境へ利用できるようにするステップと、

ｅ）前記アプリケーション構築環境において少なくとも一つの他のリファレンスに前記リファレンスをリンクし、前記リファレンスの間の関係が定義して、前記アプリケーションプログラムが実行されるとき前記関係を効果的にインプリメントするコンピュータコードが生成するリンクステップと、を備える方法。

【請求項２】　前記既存オブジェクトに対する前記リファレンスはアイコンを備え、前記カタログファシリティ及び前記アプリケーション構築環境はグラフィカル・ユーザ・インタフェースを備え、前記選択ステップは前記アイコンの一つの選択動作（selection action）をすることを備える、請求項１に記載の方法。

【請求項３】　前記既存オブジェクトへの前記リファレンスを前記アプリケーション構築環境へ利用可能とするステップは、前記カタログファシリティからアプリケーション構築ファシリティへ前記リファレンスをドラッグする（drag）ことを備える、請求項２に記載の方法。

【請求項４】　前記リファレンスはコンポーネントであり、前記カタログから前記アプリケーション構築環境内へ前記コンポーネントがドラッグされたとき、前記コンポーネントから部品（part）を生成するステップを更に含む、請求項３に記載の方法。

【請求項５】　前記リンクステップは、第１の部品と、第２の部品またはインタフェースへのリファレンスと、の間の少なくとも一つのコネクション（connection）を定義することを備える、請求項１から請求項４のいずれかに記載の方法。

【請求項６】　前記第１の部品と前記第２の部品の各々はプラグ（plug）とソケット（socket）を備え、且つ前記リンクステップは前記第２の部品上の（on）ソケットと前記第１の部品上のプラグとの間のコネクションを定義することを備える、請求項５に記載の方法。

【請求項７】　前記カタログファシリティは、前記アイコンによって表現される既存オブジェクトについての情報と前記アイコンを見る（view）ための領域（region）を備え、且つ前記リファレンスを選択するステップは少なくとも一つの前記既存オブジェクトに関係する情報の表示を引き起こすことに効果的である、請求項２から請求項５のいずれかに記載の方法。

【請求項８】　前記既存オブジェクトはインタフェースを含み、且つ前記インタフェースを編集し新しいインタフェースを定義するステップと前記既存オブジェクトのソースコードを編集し新しいオブジェクトのインプリメンテーションを定義するステップとを備える、請求項１から請求項７のいずれかに記載の方法。

【請求項９】　分散オブジェクトシステム上にインストールされるオブジェクト指向アプリケーションソフトウエアを構築するためのコンピュータにインプリメントされる方法であって、オブジェクト指向アプリケーションソフトウェアは、アプリケーション・プログラム・インタフェースと少なくとも一つの既存のオブジェクトまたは既存のオブジェクトの派生物を含み、

ａ）分散オブジェクトシステム上に提供される既存オブジェクトへのリファレンスを提供するカタログファシリティを提供するステップと、

ｂ）前記既存カタログファシリティからアプリケーションソフトウェアに含めるためのコンポーネントを選択し、且つ前記コンポーネントから部品を作る（derive）するステップと、

ｃ）前記部品は前記分散オブジェクトシステム上にある既存のオブジェクトを参照する少なくとも一つの他の部品へリンクできるアプリケーション構成環境に前記部品を利用できるようにし、それらの間の関連を定義して、それによって前記既存アプリケーションソフトウェアのためのコンピュータコードを供給するステップであって、前記アプリケーション構築環境は前記部品の間のリンクを選択し定義するためのファシリティを備え、

ｄ）前記アプリケーション構成環境において前記部品を少なくとも一つの他の部品へリンクし、それによって前記部品間の関連を定義して、前記アプリケーションプログラムが実行されるとき前記関連を効果的にインプリメントするコンピュータコードが生成するリンクステップと、を備える方法。

【請求項１０】　前記既存オブジェへの前記リファレン

スは、アイコンを備え、前記アプリケーション構成環境および前記カタログファシリティはグラフィカルユーザインタフェースを備え、且つ前記選択のステップは前記アイコンの一つの選択動作をすることを備える、請求項９に記載の方法。

【請求項１１】　前記部品を前記アプリケーション構築環境に利用できるようにするステップは、前記カタログファシリティから前記アプリケーション構築ファシリティへの前記リファレンスをドラッグすることを備える、請求項１０に記載の方法。

【請求項１２】　前記部品の前記各々はプラグとソケットを備え、前記リンクステップは１第１の部品上の（on）プラグと第２の部品上のソケットとの間のコネクションを定義することを備える、請求項９から請求項１１のいずれかに記載の方法。

【請求項１３】　前記カタログファシリティは前記アイコンによって表現される既存オブジェクトに関する情報と前記アイコンとを見るための領域（region）を含み、前記コンポーネントを選択するステップは少なくとも一つの前記既存オブジェクトに関連する情報の表示を効果的に引き起こす、請求項１０から請求項１２のいずれかに記載の方法。

【請求項１４】　前記部品は前記部品によって参照される既存オブジェクトのインタフェースへのリファレンスを含み、前記インタフェースを編集しそれによって新しいインタフェースを定義するステップと前記既存オブジェクトのソースコードを編集しそれによって新しいオブジェクトインプリメンテーションを定義するステップとを更に含む、請求項９から請求項１３のいずれかに記載の方法。

【請求項１５】　分散オブジェクトシステム上にインストールされるオブジェクト指向アプリケーションソフトウェアをコンピュータの制御のもとで構築するためのコンピュータシステムであって、オブジェクト指向アプリケーションソフトウェアは、アプリケーションプログラムインタフェースと、少なくとも一つの既存のオブジェクトまたは既存のオブジェクトの派生物と、を含み、

ａ）前記分散オブジェクトシステム上にインストールされる既存オブジェクトへのリファレンスのカタログを含むコンポーネントサービスと、

ｂ）前記リファレンスを利用してコンポジションを生成するためのコンポジションビルダであって、コンポジションビルダは前記リファレンスがリンクされそれらの関連を定義して、それによって前記アプリケーションソフトウェアのためのコンピュータコードを提供するアプリケーション構築環境を提供し、前記アプリケーション構築環境は前記リファレンスの間（among）のリンクを選択し定義するためのファシリティを備え、前記コンポジションビルダは対にされ（couple with）、

ｃ）前記コンポジションからソースコードを生成すると共にコンパイルし、それによってプログラムソースファイルを生成するコード生成ユニットと、

ｄ）前記コード生成ユニットによって生成される前記プログラムソースを処理し、前記分散オブジェクトシステム上にインストールするためのソフトウェアオブジェクトを生産するオブジェクト開発ファシリティと、を備えるコンピュータシステム。

【請求項１６】　前記リファレンスのカタログはコンポーネントを備え、前記コンポーネントは参照されているオブジェクトの特性（property）を効果的に定義し、前記特性はコンポーネントに参照されているオブジェクトのタイプ、前記コンポーネントによって参照されている前記オブジェクトによって提供されるサービス、前記コンポーネントによって参照されている前記オブジェクトのインプリメンテーション、及び前記分散オブジェクトシステム上において前記インプリメンテーションをアクセスできる名前を含む、請求項１５に記載のコンピュータシステム。

【請求項１７】　前記システムは前記コンポジションビルダに含まれるコンポーネントを識別するための選択機構を含み、前記コンポジションビルダは前記コンポーネントリファレンスを部品リファレンスに変形する（transform）ための変換機構を含み、前記部品リファレンスはコンポーネントによって参照されている前記既存オブジェクトについてのサロゲート（surrogate）である、請求項１６に記載のシステム。

【請求項１８】　前記部品はプラグとソケットを備え、前記リファレンスの間のリンクを選択し定義する前記ファシリティはプラグとソケットの間のリンクを定義するための手段含む、請求項１７に記載のシステム。

【請求項１９】　前記コンポジションビルダは部品を選択できリンクできるコンポジション・ワークスペース（workspace）を備え、コンピュータコードを編集するためのインタフェースエディタは既存オブジェクトのインタフェースを定義し、編集されるファイルを選択するためのブラウザおよびオブジェクトに関する情報を定義するためのサマリテーブルは部品によって参照されている、請求項１５から請求項１８のいずれかに記載のシステム。

【請求項２０】　前記コンポジションワークスペース、前記インタフェースエディタ、前記ブラウザおよび前記サマリテーブルの各々はグラフィカルユーザインタフェースを備える、請求項１９に記載のシステム。

【請求項２１】　前記コンポジション・ワークシート（worksheet）のための前記グラフィカルユーザインタフェースは、部品を表示し選択しリンクするためのリージョン（reigion）、選択された部品のプラグを表示し選択するためのリージョン、選択された部品のソケットを表示し選択するためのリージョン、構築されているアプリケーションのインタフェースを表示するためのリー

ジョン、既存オブジェクトのインタフェースへのリファレンスのためのリージョン、および選択された部品の属性値を設定するためのリージョンを備える、請求項２０に記載のシステム。

【請求項２２】　前記コードジェネレータと共に利用される（couple with）プログラムテンプレートのレポジトリを更に含み、前記コードジェネレータは前記コンポーネントサービスと共に利用され、前記コードジェネレータはプログラムソースファイルを生成するためのソーステンプレートと生成されたコードと結合するように設定される、請求項１５から請求項２１のいずれかに記載のシステム。

【請求項２３】　前記オブジェクト開発ファシリティはオブジェクト・アクセス・ソフトウェアのレポジトリと共に利用され、オブジェクト・アクセス・ソフトウェアの前記レポジトリは複数の分散オブジェクトと共に利用される、請求項２２に記載のシステム。

【請求項２４】　前記コンポーネントサービスは、前記コンポーネントサービスが分散オブジェクトの利用できるように、オブジェクト・アクセス・ソフトウェアの前記レポジトリと共に利用される、請求項２３に記載のシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、分散コンピューティング・システム、クライアント-サーバ・コンピューティング、及び、オブジェクト指向プログラミングの分野に関するものである。特に、本発明は、分散オブジェクトシステム上で利用されるオブジェクト指向ソフトウェア・アプリケーションの作成のための方法と装置を含む。

【０００２】

【従来の技術】ここ数年、従来のプログラミング手法を利用した場合の開発の遅延とコスト高の増加傾向にともない、オブジェクト指向プログラミングの方法論に関する関心が高まっている。オブジェクト指向プログラミング方法論は、手続きよりもデータを操作することに重点をおいている。そのためプログラマーはより直感的に実世界の問題をモデル化することができる。さらに、オブジェクトは、そのインタフェースのみを通してデータや手続きを参照することによって、プログラムのそのオブジェクト以外の部分にたいして情報を隠蔽できるように関連したデータや手続きをカプセル化する。それゆえ、オブジェクトのデータや手続きの変更は比較的プログラムの他の部分から独立している。あるオブジェクトに対する変更が他のオブジェクトのコードに影響を及ぼさないため、従来の方法を用いて書かれたコードと比べて、より容易に保守できるコードを生成できる。さらに、オブジェクトはモジュール性を備えているため、個々のオブジェクトを他のプログラムで利用することができる。このように、プログラマは、他のプログラムで何度も利用が可能な「実績がありこなれた」オブジェクトのライブラリを開発できる。信頼性の高いプログラムコードが繰り返し利用されることにより、ソフトウェアの開発時間を減少させるとともに、信頼性が増す。

【０００３】

【発明が解決しようとする課題】オブジェクト指向分散システムは、オブジェクトサーバに対するクライアントの要求を行うインタフェースを備えたオブジェクトサーバによるクライアント-サーバモデルに基づいている。そのようなシステムでは一般的に、サーバはデータと関連手続きで構成されたオブジェクトである。クライアントは、分散システムによって取り次がれる呼び出しを実行することによって、オブジェクトサーバの機能を参照することができる。オブジェクトサーバが呼び出しを受け取ると、サーバは適当なメソッドを実行し結果をオブジェクトクライアントに返す。クライアントとオブジェクトサーバは、様々な分散オブジェクトの所在をつきとめそれらの間の通信を確立する　オブジェクト・リクエスト・ブローカ（Object Request Broker:ORB）を利用して通信する。

【０００４】分散オブジェクトシステムにオブジェクト指向プログラミング方法論を導入する利益は大きいが、そのインプリメントには大きな問題がある。一般に、オブジェクトプログラミングにおいても、プログラミングの過程でソフトウェアの再利用をインプリメントする目的を達成することは困難である。典型的に、プログラマは、理解することがほとんど、もしくは、全くできないコードを積極的に利用することはない。このことは、分散オブジェクトシステムで新しいアプリケーションを開発するプログラマに対するコメントと説明を容易に付加することができない開発者にとってとくに問題である。こうして、プログラマに対して大変有用なコードが準備されているにもかかわらず、プログラマは分散オブジェクトシステムを通じてそれを十分に利用することができない場合が生じ、すでに開発されているコードを再び書いてしまう事態がおきる。

【０００５】さらに、大変大きなプログラムをコード化する過程は非常に困難である。プログラマはアプリケーションを作成するために大量の複雑なコードを解析する必要がある。オブジェクトプログラミングは、プログラマはアプリケーションのオブジェクトが備えている継承構造に気づかなければならないという、大きな課題を提示している。このように、大きなオブジェクト指向アプリケーションの開発において、プログラマはプログラム中のオブジェクトのあいだの関係について認識する必要があり、それは、吸収し解析する必要のある多量の関連文書によってよりいっそう困難なものとなる。

【０００６】異なるコンピュータ・プラットフォーム上のオブジェクトや異なるプログラミング言語で書かれた

オブジェクトを、既存のプログラミングコードを再開発したりユーザに大きな負荷をを負わせることなく、分散オブジェクトシステム上で利用できるという比較的透過的な方法によって分散オブジェクトをインストールしたり生成したりできることはプログラマやユーザにとって大変望ましいことである。さらに、分散オブジェクトプログラミングは、アプリケーションの中のオブジェクト間の関係についての解析と設計を単純化する設計や方法によってより容易なものとなる。これら両方の問題の解決は、プログラマが真に創造的な努力を必要とする部分に注目し、反復的なコーディングや解析を最小にする事を可能とし、それによってオブジェクト指向アプリケーションと分散オブジェクトアプリケーションの開発を促進する。

【０００７】

【発明を解決するための手段】本発明は、分散オブジェクトシステム上ですでに利用可能なコンピュータのコードの再利用を促進する、比較的直感的で透過的な方法で分散オブジェクト・ネットワーク上で分散オブジェクトをインストールしたり構成したりするための方法、装置、及び、コンピュータに読みとり可能な媒体を供給する。本発明の方法、装置、及び、コンピュータに読みとり可能な媒体の利用によって、分散オブジェクトの構成、設計、及び、分析が促進されることを、以下の発表と図によって明らかにする。

【０００８】本発明の一側面として、少なくとも１個の既存のオブジェクト、もしくは、既存のオブジェクトの派生物と、アプリケーション・プログラム・インタフェースを含むオブジェクト指向アプリケーション・ソフトウェアが動作する、分散オブジェクト・システム上にインストールされるオブジェクト指向アプリケーションを構成する、コンピュータにインプリメントされる方法を供給することがあげられる。具体的に、この発明の手法は、分散オブジェクト・システム上で供給される既存のオブジェクトに対するリファレンスを供給するコンポーネントを含むカタログ化機能を供給する段階を含む。カタログ化機能によってあるコンポーネントがアプリケーション・ソフトウエアの一部となるように選びだされると、その選択されたコンポーネントに参照されるオブジェクトに相当する部品は、その選択されたコンポーネントから派生する。その部品は、オブジェクト指向アプリケーション・ソフトウェアのためのコンピュータ・コードによって作られる、部品の間の関連を定義する、選択と定義リンク機能の利用によって、分散オブジェクトシステム中での、少なくとも一つの既存のオブジェクトを参照する部品とリンク付けることのできるアプリケーション構成環境の中で利用できるようになる。最後に、アプリケーション・プログラムの実行時に、部品間の関連を効果的にインプリメントできるコンピュータ・コードを生成できるような部品間の関連を定義できるアプリケーション構成環境の中で、部品は少なくとも一つの他の部品と関連付けられる。

【０００９】具体的に、既存のオブジェクトに対する参照としてアイコンがあり、カタログとアプリケーション構成環境としてグラフィカル・ユーザ・インタフェースがあり、選択の段階としてアイコンの一つを選ぶ動作がある。他の具体例として、アプリケーション構成環境で利用できる既存のオブジェクトに対する参照を作る手続きとして、カタログファシリティからアプリケーション構成機能にリファレンスをドラッグする動作がある。さらに他の具体例として、各部品はプラグとソケットを供え、リンクを作成する手続きとして最初の部品のプラグと２番目の部品のソケットとの間に連結を定義するということがある。他の具体例として、カタログファシリティはアイコンとアイコンによって表される既存のオブジェクトに関する情報をみるための領域であり、コンポーネントを選択する手続きは、選択されたコンポーネントによって参照される既存のオブジェクトに関する情報を表示を引き起こす効果がある。

【００１０】本発明のもう一つの側面は、分散オブジェクトシステムにインストールされるコンピュータ制御用オブジェクト指向アプリケーションソフトウェアのもとでの構成用コンピュータシステムを供給するということである。具体的に、この発明のシステムは、分散オブジェクトシステム上にインストールされている既存のオブジェクトシステムのリファレンスのカタログを含むコンポーネントサービスを備えている。このコンポーネントサービスはリファレンスを使って構成物を作成するためのコンポジションビルダと対である。コンポジションビルダは、オブジェクトアプリケーションソフトウェアのためのコンピュータコードを供給するために関連を定義するためにリンクできるリファレンス間の選択と定義リンクのための機能を備えたアプリケーション構成環境を供給する。プログラムソースファイルを生成する構成物からのソースコードをコンパイルし生成するためのコード生成ユニットも供給する。コード生成器はさらに、分散オブジェクトシステム上にインストールするためのソフトウェアオブジェクトを生成するためのコード生成ユニットによって生成されるプログラムソースファイルを扱うオブジェクト開発機能とともに利用される。

【００１１】具体的に、コンポジションビルダは、その中で部品を選択したりリンクしたりするためのコンポジションワークスペース、既存のオブジェクトのインタフェースを定義するコンピュータコードを編集するインタフェースエディタ、編集するファイルを選択するためのブラウザ、及び、部品に参照されるオブジェクトに関する情報を定義するためのサマリテーブルを備える。コンポジションワークスペース、インタフェースエディタ、ブラウザ、及び、サマリテーブルはそれぞれグラフィカル・ユーザ・インタフェースを備える。アプリケーショ

ン構成環境は、部品を表示・選択・リンクするグラフィカルユーザインタフェース、選択された部品のプラグを表示・選択するための領域、選択された部品のソケットを表示・選択するための領域、構成中のアプリケーションのインタフェースを表示するための領域、既存のオブジェクトのインタフェースを参照するための領域、及び、選択された部品の属性値を設定するための領域を含む。

【００１２】本発明には、分散オブジェクトシステムにインストールされるオブジェクト指向アプリケーションソフトウェアを構成するためのコンピュータプログラム製品としての側面もある。このオブジェクト指向アプリケーションソフトウェアは、少なくとも１個の既存のオブジェクトか既存のオブジェクトの派生物を含んでいる。具体的に、このコンピュータプログラム製品は、分散オブジェクトシステム上で供給される既存のオブジェクトへの参照を含むカタログファシリティを供給するための、可読なプログラムコードデバイスである。さらにこのコンピュータプログラム製品は、オブジェクト指向アプリケーションのためのコンピュータコードによって供給される相互の関連を定義するために参照をリンクできるアプリケーション構成環境を供給するためのプログラムコードデバイスをも含む。そのアプリケーション構成環境は、リファレンスの中でリンクを選択・定義するための機能を備えている。このコンピュータプログラム製品は、アプリケーション構成環境で利用可能な既存のオブジェクトへのリファレンスを選択するためのプログラムコードデバイスおも含む。さらに、アプリケーションプログラム実行時に関係をインプリメントするのに効果的なコンピュータコードが生成されるようなリファレンスの間の関係によって定義するアプリケーション構成環境の中のすくなくとも他の一つ以上のリファレンスへのリファレンスをリンクするための追加コードデバイスをも含む。

【００１３】

【発明の実施の形態】これらと他の本発明の側面、及び、利点は、以下の図を付随した詳細な記述により明らかになる。

【００１４】１．分散オブジェクトシステムの背景と物理的実装

本発明には、コンピュータシステムに蓄積されたデータを扱う様々なステップがある。これらのステップでは、物理的な量を物理的に操作する必要がある。一般に、これらの量は、その必要がないにもかかわらず、蓄積、転送、結合、比較、及び、他の操作が可能な電気的、もしくは、磁気的な信号の形をとる。共通に利用できる目的のために、これらの信号は、ビット、値、要素、変数、文字、データ構造、もしくは、同様のものとしてよく参照される。しかし、これら全て、及び、これらと同様の言葉は適切な物理量と結びつけられたものであり、また、これらの量に対する単なる便宜的なラベルであるということを覚えておかなくてはならない。

【００１５】さらに、実施される操作は、しばしば、識別する、実行する、もしくは、比較するといったような言葉で参照される。本発明の一部である、ここに記した全ての操作は、機械操作である。本発明の操作で操作を実行する有用な機械には、汎用デジタルコンピュータや同様の装置が含まれる。どの場合でも、コンピュータを操作するときの操作方法と、計算そのものの操作方法の違いについて留意しておくべきである。本発明は、他の何らかの物理的信号を生成するための電気的、もしくは、他の物理的信号に対する操作するときのコンピュータに対する操作に対する手法に関するものである。

【００１６】本発明は、これらの操作を実行する装置にも関係している。この装置は、特定の目的のために特別に作成されるかもれないし、コンピュータに搭載されるコンピュータプログラムによって再構成、もしくは、特定の動作をさせられる汎用コンピュータであるかもしれない。ここで述べる発明は特定のコンピュータ、もしくは、他の装置に関連したものではない。特に、ここで述べた方法に従って書かれたプログラムとともに様々な汎用コンピュータが使われるかもしれないし、特定の手法を実行するためにより特化した装置を作成した方が便利かもしれない。これら多様な機械に必要とされる構造については後に記す。

【００１７】さらに、コンピュータにインプリメントされた様々な操作を実行するためのプログラム命令を含む、コンピュータに読みとり可能な媒体にも関係している。その媒体とプログラム命令は、本発明のために特別に設計作成されるかもしれないし、広く一般に知られた種類のもので利用可能な、コンピュータソフトウェアの技術であるかもしれない。コンピュータに読みとり可能な媒体の例として（とくにこれらに制限される訳ではない）フロッピーディスク、磁気ディスクといった磁気媒体、CD-ROMディスクといった光媒体、MOディスクといった光磁気媒体、そして、リードオンリーメモリやランダムアクセスメモリといったプログラム命令を蓄積し実行するために作られたハードウェアなどが含まれる。プログラム命令には、コンパイラによって生成されるマシンコードとコンピュータがインタプリタを利用して実行される高級コードを含むコードとの両方が含まれる。

【００１８】分散オブジェクトシステム 10 は一般に、図１で記号的に記したオブジェクト・リクエスト・ブローカ（ORB）11 を含む。ORB 11 は、クライアントからサーバ（目的オブジェクト）にコールを配送しクライアントに応答を返すために必要な機能と、位置特定と転送機構を供給する。クライアントとサーバは同じプロセスに存在するかもしれないし、同じ機械の別のプロセスに存在するかもしれないし、全く別のマシンに存在するかもしれない。ここでの議論のために、クライアント 20

は分散オブジェクト上で操作を起動してもよいものとし、分散オブジェクトかプロセスの形をとっても、とらなくてもよいものとする。分散オブジェクトには多様な表現形式がある。例としては、分散オブジェクトはアプリケーションプログラマによて供給されるC＋＋のオブジェクトかもしれない。また、以後の2節で詳細を述べるビジュアル・アプリケーション・ビルダ 15 のなかで開発された分散オブジェクトであるかもしれない。ビジュアル・アプリケーション・ビルダは、開発者が、簡単にいうと、分散オブジェクトシステムで利用可能なオブジェクトタイプの一覧から視覚的に選択し、新しいオブジェクトを生成するためにその選択したオブジェクトによって供給される機能をグラフィカルに連結できるようにするものである。

【００１９】オブジェクト開発ファシリティ 16 は、分散オブジェクトコードの開発者のオブジェクトを覆い隠す、もしくは、カプセル化することによって、部分的に、分散オブジェクトの生成とインストールを簡略化するために利用される。オブジェクト開発ファシリティ 16 は開発者のオブジェクトをORB オブジェクトインプリメンテーションに変換するために利用される。この例では、ORBオブジェクトインプリメンテーション 14は図中で場所によってサーバとして表現されている。開発者は、ORBのインタフェースを定義するためにインタフェース定義言語を用い、オブジェクトの振るまい決めるために開発者のオブジェクトを供給し、そして、ORBオブジェクトインプリメンテーション 14 を製造するためにオブジェクト開発ファシリティ 16 を利用する。実行時に、このORBオブジェクト（サーバオブジェクト）のインスタンスはORBオブジェクトインプリメンテーション14で利用できるように生成される。オブジェクト開発ファシリティはある点でクライアントの役割を果たすオブジェクトを作るために用いてもよい点は高く評価されるべきである。

【００２０】クライアント20はサーバとスタブ（stub）21、サブコントラクト（subcontract）・レイヤ 36、フィルタ 40、そしてトランスポートレイヤ 38 を通じて連絡する。スタブ 21 はサロゲート（surrogate）22、メソッドテーブル 24、そして、スタブファンクション 25 を含む。クライアント 20は、最初に、クライアントにとってサーバに見えるサロゲート 22 と通信する。かわりに、クライアント20は、サロゲート22、メソッドテーブル24、そして、スタブファンクション（スタブ関数）25 を通じて行う代わりに、動的起動インタフェース（Dynamic Invocation interface : DII）26を通じて直接サーバオブジェクトと通信してもよい。動的起動インタフェース26はクライアントが動的要求を生成できるようにするために用いられる。

【００２１】サブコントラクトレイヤ36は、特定のサブコントラクトによって名付けられた様々なサービス（もしくは特徴・オブジェクトの機能）をインプリメントするためのサブコントラクトを利用するためにオブジェクトに必要とされる機能を提供する。サブコントラクトは、個々のオブジェクトによって起動されるであろう分散オブジェクトシステムによって提供されるサービスの質を識別する。例えば、特定のオブジェクトのために利用されるセキュリティの特徴を識別する。特定のサブコントラクトは実行時にサーバオブジェクトと動的に結びつけられる。フィルタ40は、もし利用された場合には、圧縮、暗号化、トレース、もしくは、デバッグといった様々な機能を実現する。トランスポートレイヤ 38は,クライアントとは典型的には同じプロセスを共有しないサーバとのあいだで情報を先導し、転送するための操作を行う。

【００２２】標準インプリメンテーションスイート（standard inplementarion suite）28（もしくはオブジェクトアダプタ）は、オブジェクト鍵管理と同様の方法でORBオブジェクト14と対話するサブコントラクトの集合を表す。サブコントラクトは複数のインプリメンテーションスイートに属す場合があることを特に記しておく。インプリメンテーションスイートも異なるサブコントラクトを利用する。静的スケルトン32か動的スケルトン30のいずれかの形をとるスケルトンはサーバオブジェクト78が必要とする形式に要求を変換するために利用される。このようにして、スケルトン30と32は適当なサーバオブジェクト78を呼び出す。静的スケルトン30はインタフェースに固有のオブジェクトインプリメンテーション14を呼び出すために利用され、一方、動的スケルトン30は一般にインタフェースに固有のオブジェクトが利用できない場合に使われる。ORB インタフェース 34は全てのORBのためのORBに直接つながるインタフェースで、オブジェクトのインタフェースやオブジェクトアタプタに依存しない。ORB デーモン46は、クライアントによって起動されたときにオブジェクトサーバがアクティブになっていることを保証する責任がある。

【００２３】安全プロトコル（セキュリティプロトコル）42は、インターネット上の ORB間のプロトコルを安全にする実行可能なプロトコルであり、安全な方法でトランスポートレイヤ38を通じて情報を転送するのに役立つ。インターネット上の ORB間プロトコルは異なる機械のプロセスの間で典型的に通信するためのプロトコルである。しかし、いくつかの場合、インターネット上の ORB間プロトコルは、同じマシンのプロセスの間で通信する場合もある。セキュリティサーバ54は異なる機械のプロセスの間で利用されるサービスを安全にするセキュリティ管理サーバである。

【００２４】タイプコード・Any（任意）モジュール44はタイプコード及びAny（任意）オブジェクトを実現する。タイプコードはインタフェース定義言語（IDL）のデータ型を記述し、タイプの記述がサーバとクライアン

トの間で転送されるようにする。IDL データ型のインスタンスはAny（任意）オブジェクトによってカプセル化される。Any（任意）オブジェクトはカプセル化されたデータのタイプコードとデータの汎用符号とを参照する。

【００２５】インプリメンテーションレポジトリ（implementation repository）50はオブジェクトサーバに関する情報を格納する。とくに、インプリメンテーションレポジトリ50はサーバプロセス開始するために必要とされる情報を格納する。例えば、インプリメンテーションレポジトリ50はサーバプログラムの場所、プログラムの引数、そして、プログラムに引き渡される環境変数といった情報を格納する。

【００２６】単純パーシスタンス（simple persistence）56 は、IDLに定義されたタイプが読み出せ、また、ディスクに書き込めるようにする追加コードの一部とともに、IDLコンパイラを用いて得たIDL型の実行時出力と型をインタフェース定義言語を用いて定義する。ネーミングサービス52 はORBオブジェクトに名前を付けるために利用される。クライアントは目的のオブジェクトをその名前によって検索するためにネーミングサービス52を利用してもよい。インタフェースレポジトリ（IFR）48 は分散システムじょうの全てのオブジェクトの全てのオブジェクトについて知っている。

【００２７】本発明の具体的具体例として、図２の100に記したネットワークのようなコンピュータネットワークによって互いに結合された 1台以上のコンピュータに分散オブジェクトが配置される。この図のように、ネットワーク100はネットワーク104と結合されたコンピュータ102を含む。ネットワーク104はさらに、サーバ、ルータ、もしくは、データや命令がネットワークに接続されたコンピュータの間をやりとりされるように、他のコンピュータ108、110と112に加えて106のようなコンピュータを含む。コンピュータネットワークの設計、構成、及び、インプリメントはこれらの技術と親和性がある。

【００２８】コンピュータ102、106、108、110及び112は図３の200に図示する。各コンピュータは、中央演算ユニット（CPU）もしくは並列プロセッサや分散プロセッサを含むマルチプルプロセッサのような（これらに限定されるわけではない）計算を実行するプロセッシングユニット202を含む。プロセッサ202 は、ランダムアクセスメモリ（RAM）やリードオンリーメモリといった一次記憶204 と対で利用される。RAMはプロセッサ202で現在実行されているプロセスのための、分散オブジェクトとその関連データ・操作命令を含むプログラム命令とデータを一般に格納する。ROM は、コンピュータがその機能を実現するために利用する基本的な操作命令、データ、及び、オブジェクトを一般に格納する。さらに、ハードディスク、CD-ROM、光磁気記憶装置、テープドライブといった二次記憶装置208 はプロセッサ202と双方向的に結合される。二次記憶装置208は一般にプロセッサに頻繁には利用されないプログラム命令、データ、及び、オブジェクトを一般には格納するが、プロセッサはアドレス空間の一部、例えば、仮想記憶として参照する場合もある。上記の各コンピュータは、キーボードやポインタデバイス（すなわちマウスやタブレット）といった入力媒体と一般に含む入出力源 210を備える。各コンピュータはネットワークコネクション212 も備えることができる。追加大容量記憶装置をネットワークコネクション 212を利用してCPU202に接続してもよい。上記のハードウェア及びソフトウェア要素は、ネットワーク装置と同様に、標準的な設計・構成のものであることを特に記しておく。

【００２９】ここで述べるコンピュータへのインプリメントの方法は、コンピュータシステム上のコンピュータプログラム命令を実行するための、コンピュータ科学では一般に知られた技術や装置を利用してインプリメントするものである。ここでは、コンピュータシステムとは、RAM、ROM、CD-ROM、やハードディスクといった、プロセッシングユニットによって、データや命令を交換する一つ以上のデータ記憶装置とともに、データや命令を取り扱うためのプロセッシング装置（中央演算装置など）を含む。データ記憶装置は専用（プロセッシングユニットと直接結合されている）かリモート（コンピュータネットワークを介してプロセッシングユニットと結合されている）である。コンピュータネットワークを介してプロセッシングユニットと結合されるリモートデータ記憶装置はプログラムを特定のワークステーション上のプロセッシングユニットで実行するためにプログラム命令を送るために利用できる。さらに、プロセッシング装置は、同じ物理機構か（並列プロセッサの場合）か、ネットワーク（分散プロセッサの場合）かのいずれかを介して 1個以上のプロセッシング装置を接続できる。リモートで結合されたデータ記憶装置とプロセッサの利用はコンピュータ科学の中でより一般的なものになるであろう。ここではコンピュータネットワークとは、互いに通信できるコンピュータシステムの集まりが相互に結合した通信チャンネルの集まりのことをいう。通信チャンネルには、ツイストペアケーブル、同軸ケーブル、光ファイバ、衛星通信、もしくは、デジタル無線といった通信媒体を含む。コンピュータシステムは、広い地域（16～1600kmの場合。WAN）に分散していてもいいし、局所的なネットワーク（3～50mの場合。LAN）でもよい。さらに、さまざまなローカル及びワイドエリアネットワークは、集合的なコンピュータネットワークへ結合することができる。そのようなコンピュータネットワークの連合の例としてインターネットがある。

【００３０】３．本発明の構成ツール

図４の400に、分散オブジェクトシステムでオブジェクト指向アプリケーションを構成するシステムを示す。本

発明のこのシステムは。図３で示し先に述べたような分散オブジェクトシステムにインストールするアプリケーションを構成することが、ユーザ、典型的にはプログラマ、にできるようにするコンポジションビルダ402 を含む。コンポジションビルダは、分散オブジェクトシステム上で利用可能なオブジェクトを、ユーザやプログラマが利用できるようにするコンポーネントサービス404 と共に利用される。具体的に、このような利用ははカタログによって可能となる。カタログとは、簡単にいって、分散オブジェクトシシステム上でプログラマが利用できるソフトウェア資源の一覧である。カタログは具体的に、カタログに含まれているコンポーネントに参照されるオブジェクトに関するインプリメンテーションと機能に関する情報をプログラマに提供する。このように、カタログは、ユーザにソフトウェア資産を利用可能にすることによって分散オブジェクトシステムにわたる協調作業を促進し、利用可能なオブジェクトとソフトウェアに関する詳細な情報を提供することによってソフトウェア開発者間での共同作業を促進する。コンポジションビルダ 402はさらに、コンポジションビルダによって作られたコンポジションを格納するプログラムテンプレートレポジトリ402と共同して410にしめすようなプログラムのソースファイルを製造するコードジェネレータ408と共に利用される。

【００３１】プログラムソースファイル410は次にODFコンパイラ・リンカ414に送られる。さらに、ODFコンパイラ・リンカ414は、コンポーネントサービス404と共に利用されるオブジェクトアクセスソフトウェアと共に利用される。オブジェクトアクセスソフトウェア412は、分散オブジェクトを構築するために OMG CORBA仕様に従うようにIDLインタフェースについての言語の対応を準備するためにIDL コンパイラが利用されたときに生成されるスタブの集合からなる。ODFコンパイラ・リンカは、420に概略を示した、順次ネットワークオブジェクト422を利用するオブジェクトソフトウェア416とネットワークソフトウェア 418の両方を生成する。これらのネットワークオブジェクトは、コンポーネントサービス404によってODFコンパイラ・リンカ414に供給されるオブジェクトアクセスソフトウェア414を用いて利用される。

【００３２】本発明は、分散オブジェクトシステム中に配置できる 1個以上の既存のオブジェクトかその派生物からソフトウェアアプリケーションがその中で構成される非常に直感的で便利な環境をプログラマやユーザに提供することによってコードの再利用しプログラマの生産性をあげるという上記の目的を促進するコンポジションユーザインタフェースを含む。本発明では、コンポジションビルダインタフェースは、オブジェクト指向ソフトウェアの設計を簡略化し既存のコードの再利用を促進するように設計されたいくつかのビューを含む。最初のビューを図５の500に示す。このビューには、タイトルバー 502とコントロールリージョン503から成るウィンドウ501で構成されるコンポジション構成環境が示されている。コントロールリージョン 503は、複数のインタフェースモード制御ボタン504と作成ボタン505とコンパイルボタン506を含む。

【００３３】インタフェースツールリージョン507は、一般に510に示されるような（インタフェースはそれぞれ「App」と「Stream」である）現在のインタフェースが起源とする既存のインタフェースを表すアイコンに加えて、現在操作されているインタフェース（たとえば 509のAudioAppと書かれたインタフェース）を表すアイコンを表示するためのカレントインタフェース・リージョン508を含めて表示する利用可能なインタフェースの表示を含む。開発中の特定のアプリケーションのためのユーザインタフェースを調べて開発するためのエディタを起動するユーザインタフェースコントロールボタンを 511に示す。こようなユーザインタフェースを作成し編集するユーティリティはコンピュータ科学技術としては一般的なものである。

【００３４】開発の過程で操作される部品の様々なパラメータをプログラマが制御できるようにする値編集リージョンは一般に 512のようなものである。512 の表示は、操作中の部品（ここではオブジェクトAudioDev530） を示す最初の領域 514、編集中の部品の特性（ここではVolumeという特性）のためのフィールド、及び、編集中の特性の値（ここでは特性Volumeの値が10.0に設定されている）を示す値フィールドを含む。

【００３５】エディタ500はさらに、522に示した編集可能なSocket_1といった、構成中のアプリケーションのインタフェースのために供給されているソケットのための領域520を含む。同様に、2番目のフィールド524はアプリケーションによって供給されるPlug_1 526といったプラグを表示する。中央の部分 525は、構成中のアプリケーションをインプリメントするためのコードを定義するためにグラフィカルに様々な既存のオブジェクトが並べられて連結されたものとして表される部品がしめされるワークシートである。ワークシートには二つの既存のオブジェクト（AudioDev 530とInputStream532）を表す部品が示されている。部品530はコネクション534によってSocket_1と連結されており、さらにコネクション 536で InputStream 部品 532 と連結されている。InputStream 532もインタフェース540とPlug_1 526とコネクション538によって連結されている。

【００３６】コネクションは、544に示したプラグと546に示したソケットを利用して、部品と他の部品の間もしくはインタフェースの間に作られる。ソケットは、このオブジェクトから他の要求元オブジェクトに渡されるオブジェクトリファレンスを構成する、このオブジェクトによって供給されるサービスを表す。プラグは、逆に、オブジェクトが要求したり処理したりできるサービスを

表す。これらのことはオブジェクトプログラミング技術の分野ではよく知られたもので、最初のオブジェクトのプラグと２番目のオブジェクトのソケットの間のコネクション（コネクション536のような）を描くことによって通信が図示されたオブジェクトリファレンスを送ったり操作したりすることで、オブジェクトはそれらの間で通信する。図５で示したインタフェースから明らかなように、プラグとソケットの間のコネクションを定義する対話的ツールを用いて連結されたオブジェクト（すなわち部品）を表すアイコンを提示するグラフィカルな環境によるプログラミングオブジェクトコネクションパラダイムの利点によってアプリケーションの開発が促進される。図５には、550で概略をしめしたコンポーネントカタログも示されており、カタログは522に示すコンポーネントを含む。

【００３７】本発明の実施例では、コンポーネントカタログ550のコンポーネント552のようなコンポーネントを選択すし、そのコンポーネントをコンポーネントカタログからワークシートへドラッグすると、コンポーネントは552'のコンポーネント部品に変形される。さきに述べたように、部品は、部品とコンポーネントの両方を参照するオブジェクトタイプのための入れ物である。部品は、それに相当するオブジェクトに利用可能なプラグやソケットを差し示す。ワークシート中の552'のような部品のプラグとソケットを連結することによって。本発明のコンポジションビルダは部品の間の必要なコネクションを確立したものに相当するコードを生成することができ、このように分散システムにわたって適当なオブジェクトを配置するという困難なプログラマの仕事を軽減し、これらのオブジェクトを利用するために必要なボイラープレートコードを供給し、そして、これらのオブジェクトの間の通信を確立するために必要な文法上の適当な引数を定義する。このようにコンポジションビルダは、分散システム上で利用できるコードを参照し再利用するより簡単な方法を提供し、アプリケーションを構成するための直感的な枠組みに供給されるコードをインプリメントすることにより、分散オブジェクトシステムにインストールされるアプリケーションの構成を促進する。このように、ワークシートを利用することによって、もし必要ならば容易に早く評価・修正することができるコードが設計できるような現在のコンポジションを構成しているオブジェクトの間の関連を容易にプログラマは識別できる。このように、本発明のコンポジションワークシートはアプリケーションのインプリメンテーション同様その設計をも促進する。

【００３８】２番目の状態では、本発明のコンポジションビルダの、上記のボタン504の一つを押したときに利用される２番目のインタフェースが提供される。この状態では、コンポジションビルダは、図６の600に示すようなファイル閲覧ユーティリティを含む。プログラムの構成時に利用するファイル（例えばプログラムのソースファイルや実行可能コードライブラリ）をプログラマに比較的便利にみれるようにすることによってオブジェクトアプリケーションの構成を促進するファイル閲覧ユーティリティ 600はコンピュータプログラミング技術の分野では有用なものである。構成中のコンポジション参照され操作ている継承されたインタフェースや部品を用いるために必要な全てのオブジェクトアクセスソフトウェア（412）の一覧が操作されるのと同様に、コンポジションビルダ402とクライアントコンポーネントサービス404とによって表示も操作される。

【００３９】600で示したように、ブラウザは、タイトルバー602、ワークシート500のコントロールリージョン503とほぼ同じコントロールリージョン603、及び、ファイルファイル検索・閲覧リージョン608から成るウィンドウ601を含む。検索・閲覧リージョンは、おされたときに、様々な特徴（例えばテキスト文字列）に基づいてファイルやオブジェクトを検索するために、コンピュータ科学技術の分野では一般的なサーチエンジンを用いて検索する事のできるファイルを識別するものやオブジェクトの特徴をユーザに示す検索（Find）ボタン610を含む。検索ボタン610にはファイル検索エンジンに情報を入力するためのウィンドウ612 がある。Patternウィンドウファイル614とContainsウィンドウファイル 616は両方とも、検索ボタンを押すことによって開始される検索機能を実行するための追加属性を入力するためのものである。ウィンドウ600のリージョン618は様々なファイルを階層的に表示するための三つの行 620、622、及び、624を含む。ファイルの階層構図の中でより上位のものがコラム 620のより左の方に表示される。そのようなファイルには、626のotherや、上下の平行線によって選択されていることが示されている628のsourceといったファイルが含まれる。もしファイルがsourceディレクトリに含まれているときには、630 のファイルsample.ccようにコラム622の中に一覧が表示される。より高次のディレクトリにサブディレクトリがあるところにはコラム622にサブディレクトリが620に示したのとほぼ同じ形式で表示される。サブディレクトリを選択すると（ここには示されていない）選択されたサブディレクトリにあるファイルとサブディレクトリがコラム 624に表示されるようになる。これらの分野ではごくふつうの技術なのだが、コラム 624に表示されているサブディレクトリをさらに選択すると、コラム620の内奥がコラム 622のものと置き換えられ、コラム622に表示されているものが623に表示され、もともと624にもともとあるサブディレクトリの内容がコラム624に表示されるという再配置が生じる。

【００４０】本発明のもう一つのコンポジションビルダインタフェースの状態では、選択された特定の部品もしくはコンポーネントに関する情報を入力するための 3番

目のインタフェースが提供される。図7の700に示すように、タイトルバー702と、図5と図6で示したものとほぼ同じ機能を実現するボタン704、705と706を備えたコントロールリージョン703とから成るウィンドウからこのインタフェースは構成される。表示領域708には、他のコンポジションのオブジェクトと含めるために構成されたオブジェクトの性質や特性を他のユーザのカタログが決定できるようにするリファレンスオブジェクトの性質をプログラマが記述できるような、特定のオブジェクトやコンポーネントに参照されるオブジェクトの識別や記述に関する様々な情報が提供される。そのような情報には、コンポーネントの名前、600 のブラウザを用いて検索できる様々なキーワード、選択されたコンポーネントや部品に関連する様々なコンポーネント、選択されたコンポーネントや部品に関連する様々なコンポーネント、コンポーネントや部品に参照されるオブジェクトの状態、現在選択されているコンポーネントや部品を現在利用している様々なコンポーネント、参照されているオブジェクトに関する様々な構造化された情報へのリンクを定義する例やテスト、更新履歴と同様の作者の身元や住所などが含まれる。さらに、710 に示すような目的や機能に関する短い記述が提供される。このように、このインタフェースが、分散システム上にすでにインプリメントされたオブジェクトの再利用を促進するための関連情報を比較的素直にプログラマに提供する利点がわかる。

【００４１】もう一つの状態では、本発明のコンポジションビルダは、構成中のアプリケーションを使うための様々なインタフェースを修正するためのエディタを含む。そのようなエディタの一例を図8の800に示す。インタフェースエディタは、タイトルバー802と、図5で示したのとほぼ同じ機能を備えたボタン804、805と806を含むコントロールリージョンとを含むウィンドウ801から構成される。編集ポート808では、インタフェースを定義し編集するためにテキストを入力することができ、それに加えて IDLコンパイラといった関連するコンパイラからのメッセージを受け取ることができる。エディタはC＋＋といったオブジェクトコードとIDLコードの両方を編集するための標準的なテキストエディタである。

【００４２】上記のインタフェースは、コンピュータ科学の技術、より限定して、コンピュータインタフェースの技術の分野では知られたソフトウェアや技術を用いて構成されている。上記のインタフェースから、「OpenStep」（Sun Microsystems ofMountain View,CA より）、「Windows」（Microsoft Corporation of Redmond,WA より）、Macintoshオペレーティングシステム（Apple Computer of Cupertin, CAより）、そして「X Windows」（X Consortium of Cambridge, MA）のソフトウェアを構成できる。本発明から出発することなく上記のインプリメンテーションに様々な変更を行うことができる点は優れている。

【００４３】コンポジションビルダとその関連インタフェースについて述べてきたが、つぎにここで述べたコンポジションビルダを用いるアプリケーションの構築について述べる。そのような方法の一つを図9の900に示す。902で始まり、新たな構成は904で作成される。906では、構成するアプリケーションに必要とされる情報と供給される情報を提供するインタフェースを IDLがインプリメントすることで、構成のためのインタフェースが選択される。908 では、オブジェクトに対するインタフェースがすでに存在しているかどうかについての決定が行われる。もしそのようなインタフェースが存在しないならば、上記のワークシート500こコンポジションビルダ、とくに、ワークシート500の領域 520と524を用いて、分散オブジェクトシステム上の他のオブジェクトと通信するために作られるアプリケーションに必要なプラグとソケットが910 で作られる。もしインタフェースがすでに存在しているか、910で新しいインタフェースのためのプラグとソケットが定義されたならば、新しいコンポジション（すなわち、開発中のアプリケーション）の実装で用いられるコンポーネントが912 で選ばれる。これらのコンポーネントはカタログ550から選ばれ、552に表示されるコンポーネントとして構成される。

【００４４】先に示したように、コンポーネント 552のようなコンポーネントを表現するアイコンを選択すると、プログラマにはそのコンポーネントに関する情報が表示される。ステップ914では、図5の534、536、538と540で示したようなコネクションを用いて、ワークシートの領域525に配置されたさまざまなIDLインタフェース、プラグ、及び、他の部品のコンポーネントが連結される。ステップ916 では、関連するオブジェクトが機能するようにメソッドが実装される。これらの実装は図8の800で示したようなテキストエディタを用いて行われる。この実装のち、ソースコードが生成され、副ジェネレータ408によってコンパイルされる。ステップ920でいったんソースが生成されコンパイルされると、生成されたコードの必要なデバッグを実行するために必要なテストを行うことができ、ステップ922と924で最終的なコードがパッケージ化されインストールされる。920から924にわたるステップは、図4の414のようなオブジェクト開発ファシリティ（ODF）と連携して行われる。

【００４５】図１０の1000に新しいインタフェースを構築する方法を示す。1002で開始され、ステップ1004で新しいインタフェースに名前が付けられ、ステップ1006では、有効なインタフェース名に対する評価項目と与えられた名前の文法を比較する標準的な方法を用いて確認する。ステップ1008では、与えられた名前が有効かどうかの決定を行う。もしその名前が有効でないならば、新しい名前が与えられるステップ1004へ戻るか元の名前をを

修正する必要がある。一般に、このように流れの制御を戻ると同時にプログラマに対して警告が表示される。1008でいったん有効な名前が与えられるとユーザは1010で1個以上のベースインタフェースを選択できる。しかし、必ずベースインタフェースを選ぶ必要があるわけではない。ベースインタフェースとしては、構成されるインタフェースにふさわしいインタフェースに最も近いものを提供しているものが選ばれるべきである。1012は、プラグとソケットは、図5の500に示されたインタフェースと上記の600、700と800の関連インタフェースと共に用いて編集される。

【0046】ステップ1014では、インタフェース定義言語を用いて定義されたインタフェースをこの段階でIDLテキストを直接入力したいとプログラマが思ってもよい。この編集作業は、図8の800に示したようなエディタを用いて実現される。ステップ1016では、全ての編集作業はすべてIDLコンパイラといったものを利用して確認され、その全ての結果がユーザに表示される。もし編集が1018で有効ならば、ベースインタフェースの表示と関連するプラグとソケットがステップ1020で更新される。もしそうでない場合には、さらなる編集がステップ1014で行われる。典型的に、もし編集が有効でないならば、ユーザは図8でしめされたような様々なデバッグ用メッセージがユーザに対して表示されている。ステップ1020のプラグとソケットの更新に引き続いて、ステップ1022で編集作業は終了する。

【0047】プラグとソケットの間のコネクションは、図11の1100に示すようなブラウザを用いて決定される。ブラウザ1100には、1102に示すようなプラグのためのものを含む様々フィールドや1106、1110と1114に示すようなソケットのためのものを含む様々なフィールドが含まれる。特定のプラグ、すなわちフィールド1104のプラグ 2、を選択すると、そのプラグに関連付けることのできるソケットがフィールド1108に表示される。上に述べたものは図5の600に示したブラウザに関するもので、ソケットはそれ自身別のソケットと関連があり、フィールド1112と1116には、選択されたソケットに関係されたソケットも表示される。例えば、この図では、ソケット2は1118 で選択されその関連ソケット13は1120で選択されたものとなり、フィールド1116でソケット14、15と16の表示がユーザに提供されている。1122に示すようなソケット 15の選択はプラグ2がソケット15とリンクされていることを示している。

【0048】上記の発明の記述は理解を明確にするためにある程度詳細に述べたものだが、追加の主張の範囲で、ある程度の変更と修正を実施してよいことがわかる。例えば、あるインタフェース設計に仕様が仕様が記述されているが、他のインタフェース設計も同様に利用できる。例えば、図6から図8には1個以上のコントロールボタンや入力フィールドが示されているが、これらは本発明の範囲や主旨からはずれることなく取り除くことができる。さらに、ここで述べたインタフェース設計に、オブジェクトの間のコネクションを特定する様々な方法を導入することができる。例えば、コンポーネントや部品以外のオブジェクトは既存の分散オブジェクトを表現するために利用することができる。

【図面の簡単な説明】

【図1】図1は、本発明に従うのオブジェクト・リクエスト・ブローカ（ORB）の概略図である。

【図2】図2は、本発明に従うコンピュータ・ネットワークの構成図である。

【図3】図3は、本発明に従うコンピュータ・システムのブロック図である。

【図4】図4は、本発明に従う、分散オブジェクトシステムでオブジェクト指向アプリケーション構築のためのシステムの概略図である。

【図5】図5は、本発明に従うコンポジション・ビルダの図である。

【図6】図6は、本発明に従うのファイル・ブラウザの図である。

【図7】図7は、本発明に従うコンポーネントサマリシートとエディタの図である。

【図8】図8は、本発明に従うのインタフェース定義言語（IDL）エディタの図である。

【図9】図9は、本発明に従う分散オブジェクトシステム上に複合オブジェクトアプリケーションを配置する手続きの流れ図である。

【図10】図10は、本発明でのオブジェクトインタフェース構築の流れ図である。

【図11】図11は、本発明に従うソケットとプラグの間の連結を定義するブラウジング・ユーティリティの図である。

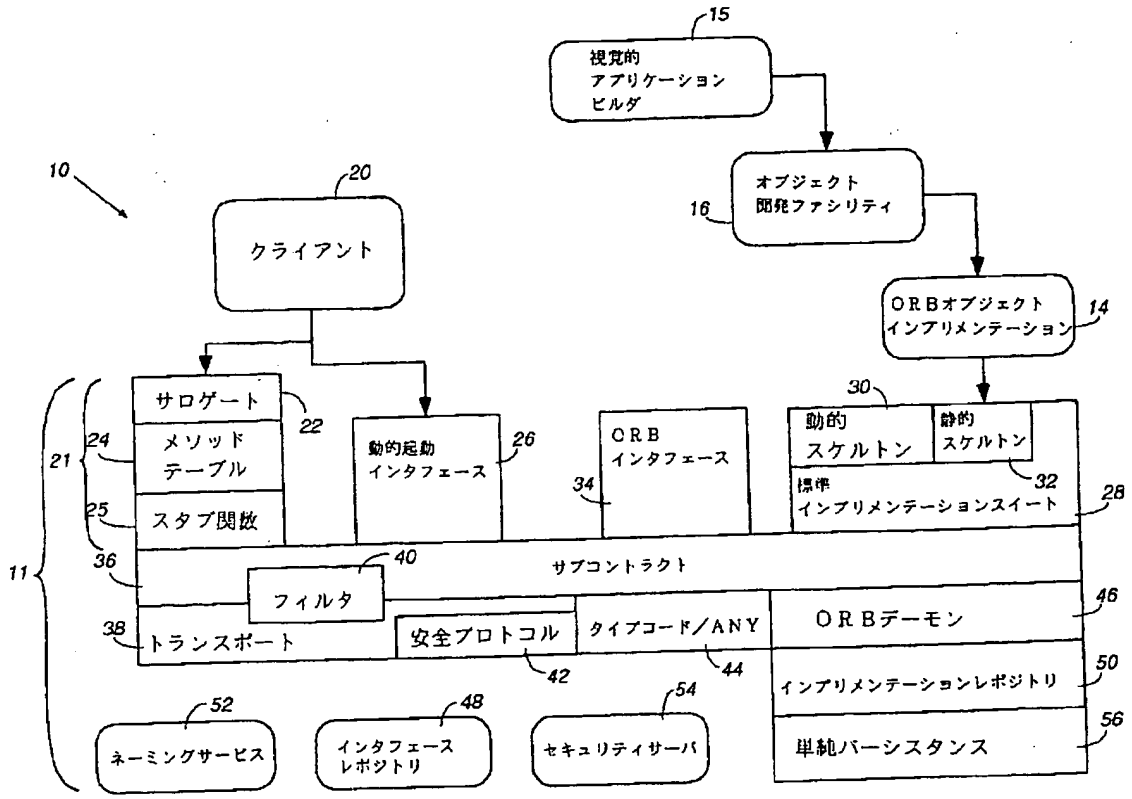
【符号の説明】

10…分散オブジェクトシステム、14…ORBオブジェクト・インプリメンテーション、16…オブジェクト開発ファシリティ、20…クライアント、22…サロゲート

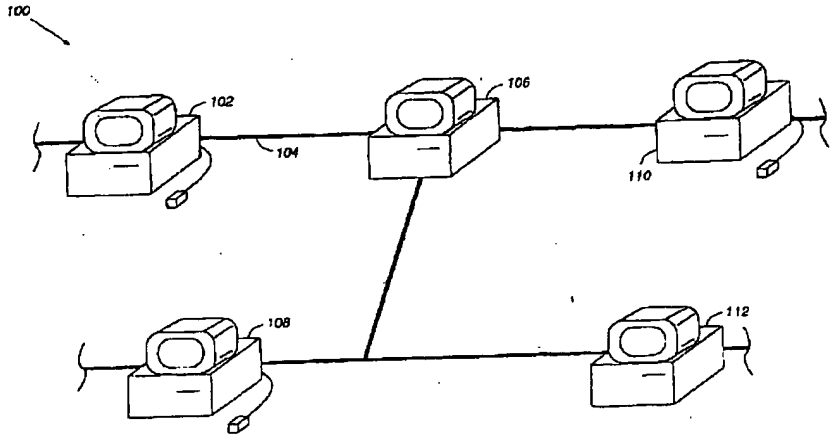
24…メソッド表（メソッド・テーブル）、25…スタブ関数、26…動的起動インタフェイス、28…標準インプリメンテーション・スイート

30…動的スケルトン、32…静的スケルトン、34…ORBインタフェイス、36…サブコントラクト、38…トランスポート層、42…安全プロトコル、44…タイプコード／ANY、46…ORBデーモン、48…インタフェース・レポジトリ、50…インプリメンテーション・レポジトリ、52…ネーミング・サービス、54…セキュリティ・サーバ、56…単純パージステンス

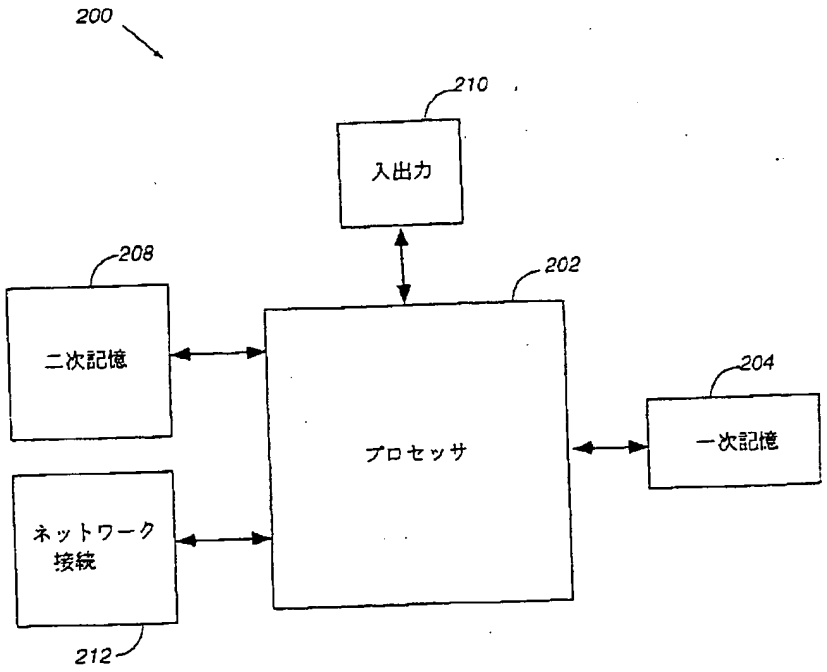
【図1】


15
視覚的
アプリケーション
ビルダ
10
20
クライアント
16
オブジェクト
開発ファシリティ
ORBオブジェクト
インプリメンテーション
14
30
サロゲート
22
24
メソッド
テーブル
21
25
スタブ関数
動的起動
インタフェース
26
ORB
インタフェース
34
動的
スケルトン
静的
スケルトン
32
標準
インプリメンテーションスイート
28
11
36
サブコントラクト
40
フィルタ
38
トランスポート
安全プロトコル
42
タイプコード/ANY
44
ORBデーモン
46
インプリメンテーションレポジトリ
50
52
ネーミングサービス
48
インタフェース
レポジトリ
54
セキュリティサーバ
単純パーシスタンス
56

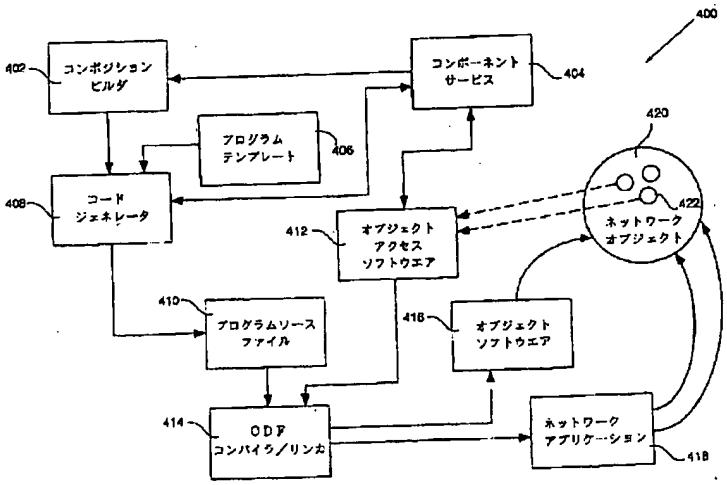
【図2】


100
102
104
106
110
108
112

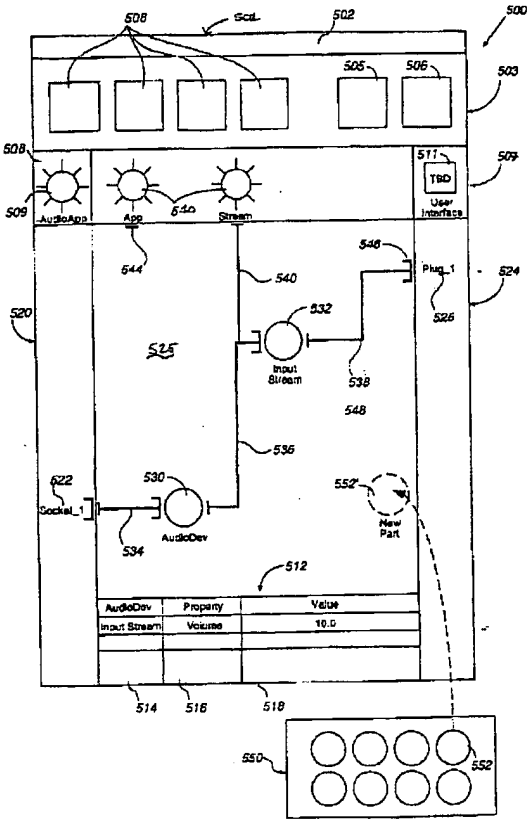
【図3】


200
210
入出力
208
二次記憶
202
プロセッサ
204
一次記憶
ネットワーク
接続
212

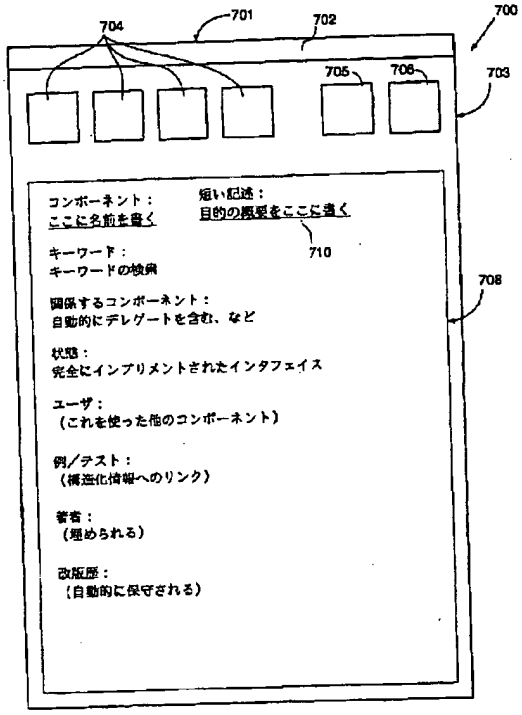
【図4】


400
402
コンポジション
ビルダ
コンポーネント
サービス
404
420
プログラム
テンプレート
406
408
コード
ジェネレータ
422
ネットワーク
オブジェクト
412
オブジェクト
アクセス
ソフトウエア
410
プログラムソース
ファイル
416
オブジェクト
ソフトウエア
414
ODF
コンパイラ／リンカ
ネットワーク
アプリケーション
418

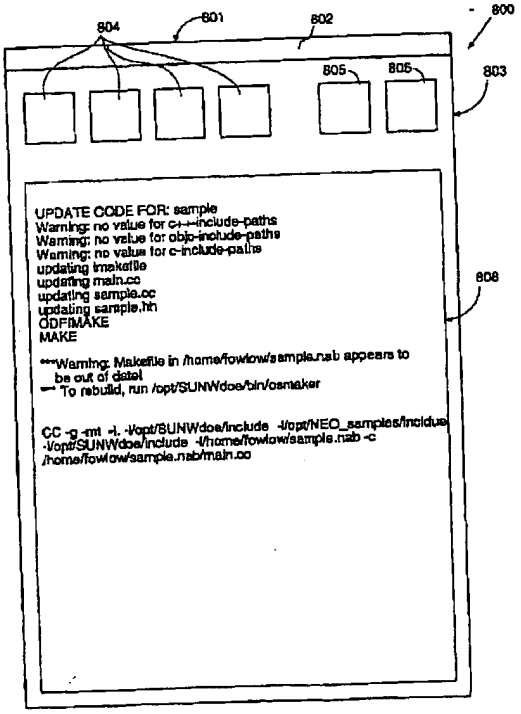
【図5】


500
502
503
504
505
506
507
508
509
511
TBD
User Interface
AudioApp
App
Stream
540
544
546
Plug_1
524
526
520
525
532
Input Stream
538
548
536
622
Socket_1
530
AudioDev
534
552'
New Part
512
AudioDev
Property
Value
Input Stream
Volume
10.0
514
516
518
550
552

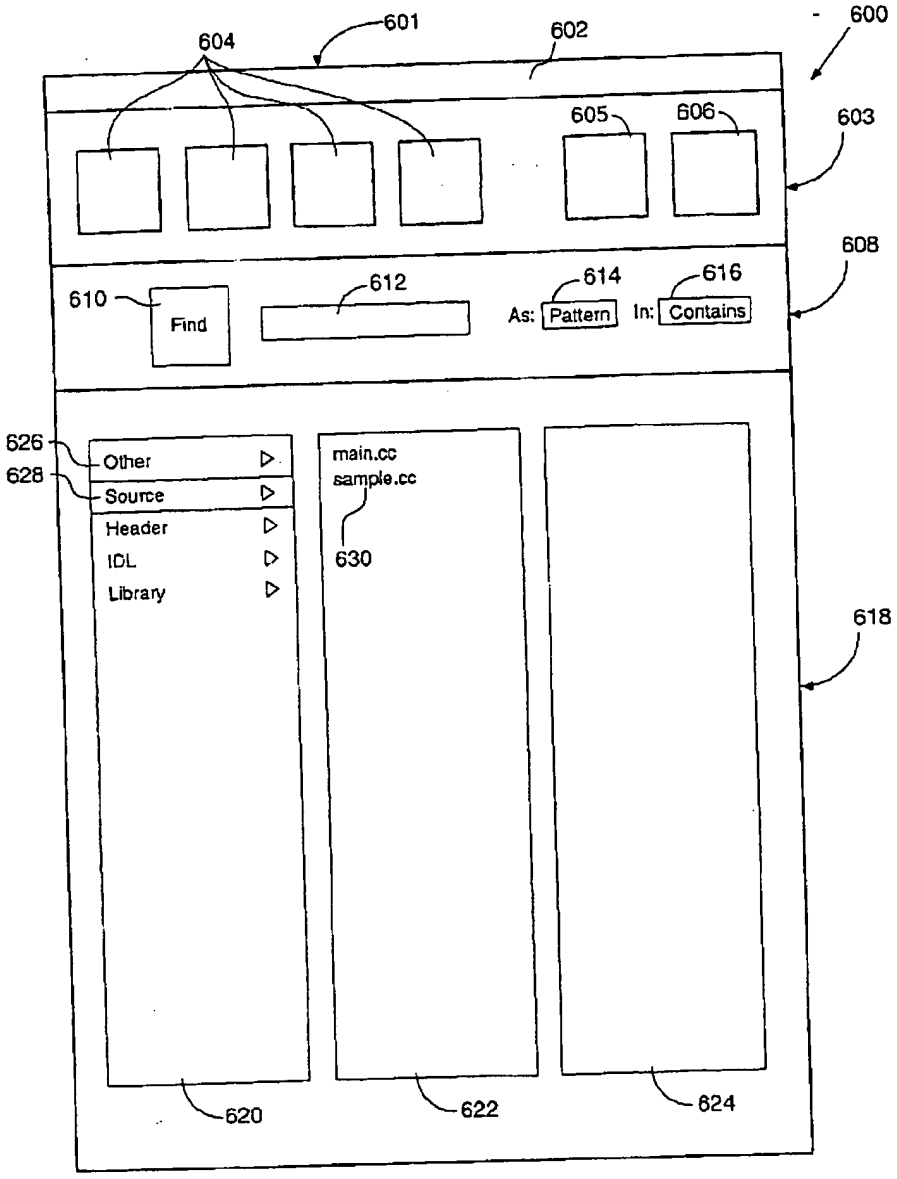
【図7】


700
701
702
703
704
705
706
708
710
コンポーネント：
ここに名前を書く
短い記述：
目的の概要をここに書く
キーワード：
キーワードの検索
関係するコンポーネント：
自動的にデレゲートを含む、など
状態：
完全にインプリメントされたインタフェイス
ユーザ：
（これを使った他のコンポーネント）
例／テスト：
（構造化情報へのリンク）
著者：
（埋められる）
改版歴：
（自動的に保守される）

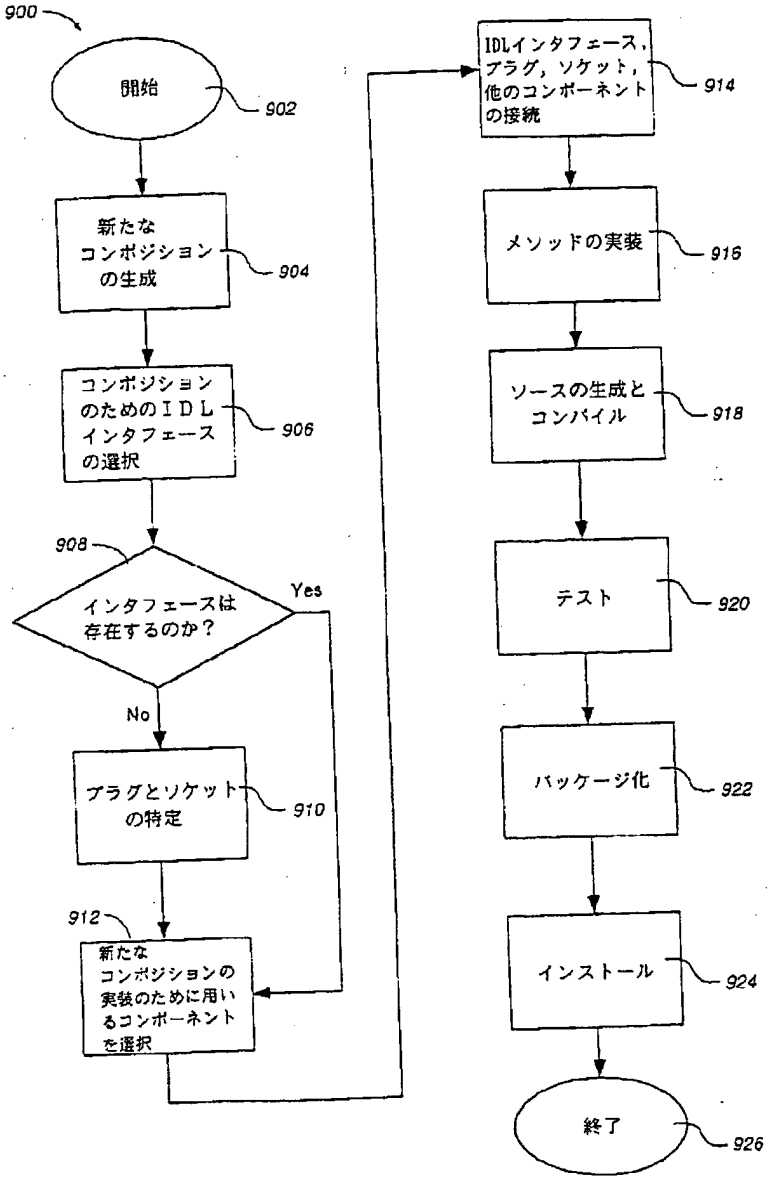
【図8】


800
801
802
803
804
805
806
UPDATE CODE FOR: sample
Warning: no value for c++-include-paths
Warning: no value for objc-include-paths
Warning: no value for c-include-paths
updating Imakefile
updating main.cc
updating sample.cc
updating sample.hh
ODFIMAKE
MAKE
***Warning: Makefile in /home/fowlow/sample.nsb appears to be out of date!
*** To rebuild, run /opt/SUNWdoe/bin/osmaker
CC -g -mt -I. -I/opt/SUNWdoe/include -I/opt/NEO_samples/include -I/opt/SUNWdoe/include -I/home/fowlow/sample.nsb -c /home/fowlow/sample.nsb/main.cc

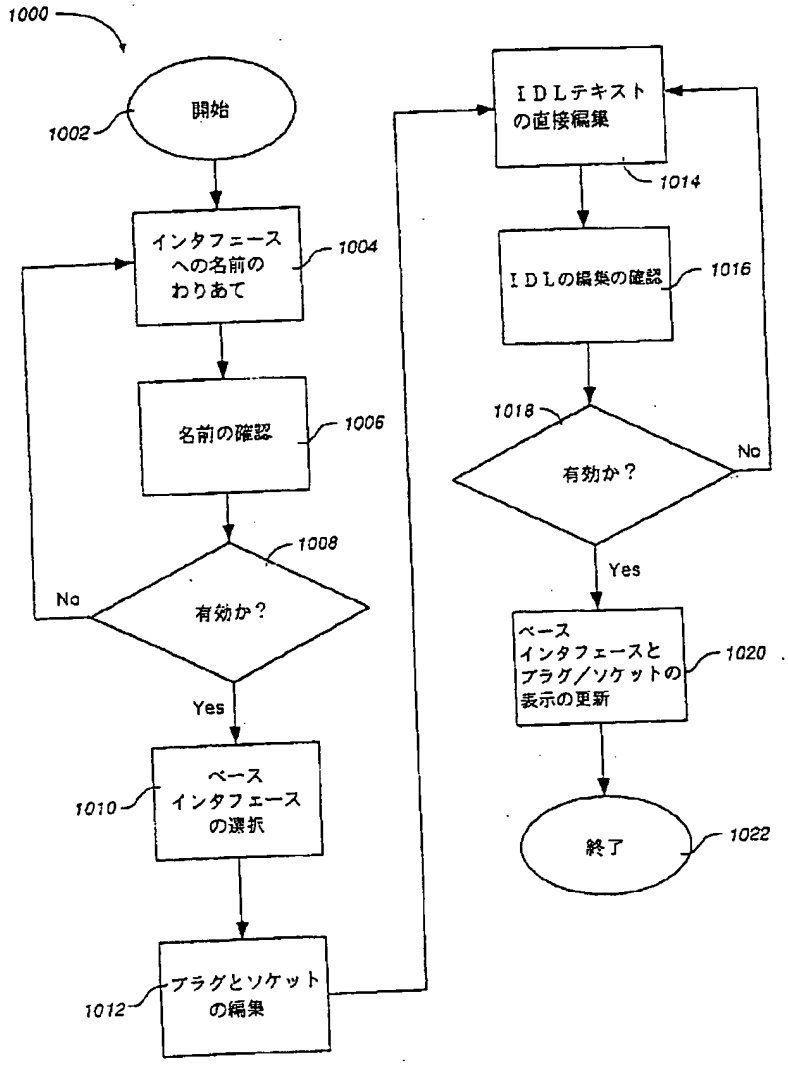
【図6】


600
601
602
604
603
605
606
608
610
Find
612
614
As:
Pattern
616
In:
Contains
626
Other
628
Source
Header
IDL
Library
main.cc
sample.cc
630
618
620
622
624

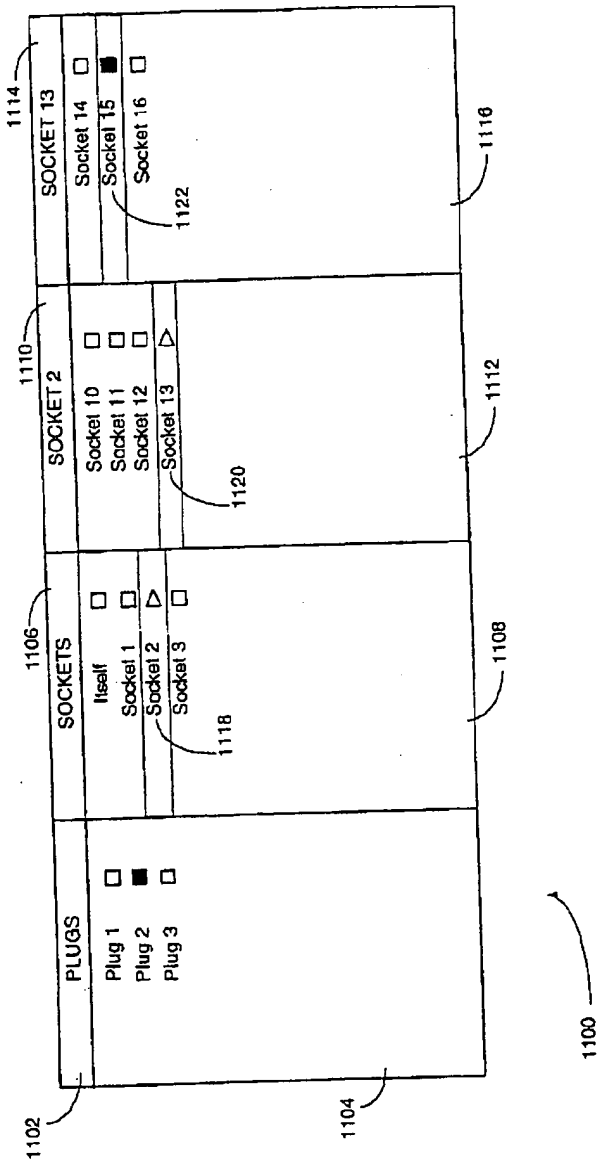
【図9】


900
開始
902
新たな
コンポジション
の生成
904
コンポジション
のためのＩＤＬ
インタフェース
の選択
906
908
インタフェースは
存在するのか？
Yes
No
プラグとソケット
の特定
910
912
新たな
コンポジションの
実装のために用い
るコンポーネント
を選択
IDLインタフェース，
プラグ，ソケット，
他のコンポーネント
の接続
914
メソッドの実装
916
ソースの生成と
コンパイル
918
テスト
920
パッケージ化
922
インストール
924
終了
926

【図10】


1000
開始
1002
インタフェースへの名前のわりあて
1004
名前の確認
1006
有効か？
1008
No
Yes
ベースインタフェースの選択
1010
プラグとソケットの編集
1012
IDLテキストの直接編集
1014
IDLの編集の確認
1016
1018
有効か？
No
Yes
ベースインタフェースとプラグ／ソケットの表示の更新
1020
終了
1022

【図11】



フロントページの続き

(72)発明者　グレゴリー　ビー．　ニュイエンズ
アメリカ合衆国，　カリフォルニア州，
メンロ　パーク，　ローレル　アヴェニュー　403

(72)発明者　フランク　ルドルフ
アメリカ合衆国，　カリフォルニア州，
サラトガ，　ユニヴァーシティ　ドライヴ　241